# Utilizing Typed Dependency Subtree Patterns for Answer Sentence Generation in Question Answering Systems

**Rivindu Perera · Parma Nand · Asif Naeem**

Received: date / Accepted: date

**Abstract** Question Answering over Linked Data (QALD) refers to the use of Linked Data by question answering systems and in recent times this has become increasingly popular as it opens up a massive Linked Data cloud which is a rich source of encoded knowledge. However, a major shortfall of current QALD systems is that they focus on presenting a single fact or factoid answer which is derived using SPARQL (SPARQL Protocol and RDF Query Language) queries. There is now an increased interest in development of human like systems which would be able to answer questions and even hold conversations by constructing sentences akin to humans. In this paper we introduce a new answer construction and presentation system, which utilizes the linguistic structure of the source question and the factoid answer to construct an answer sentence which closely emanates a human generated answer. We employ both Semantic Web technology and the linguistic structure to construct the answer sentences. The core of the research resides on extracting dependency subtree patterns from the questions and utilizing them in conjunction with the factoid answer to generate the answer sentence with a natural feel akin to an answer from a human when asked the question.

We evaluated the system for both linguistic accuracy, and naturalness using human evaluation. These evaluation processes showed that the proposed approach is able to generate answer sentences which have linguistic accuracy and natural readability quotients of more than 70%. In addition, we also carried out a feasibility analysis on using automatic metrics for answer sentence evaluation. The results from this phase showed that the there is not a strong correlation between the results from automatic metric evaluation and the human ratings of the machine generated answers.

**Keywords** Answer Presentation · Question Answering · Dependency Parsing · Linked Data · Semantic Web

R. Perera
SECMS (D-58), Auckland University of Technology,
Auckland 1010, New Zealand
E-mail: rivindu.perera@aut.ac.nz

P. Nand · A. Naeem
SECMS (D-75), Auckland University of Technology,
Auckland 1010, New Zealand

## 1 Introduction

Question Answering over Linked Data (QALD) transcends the traditional Question Answering (QA) process to a new dimension. Instead of a corpus or any other text collection, QALD employs the massive Linked Data cloud as an information source to retrieve the answers to a question posed in a natural language [15]. The first step in QALD is to reform the natural language question as a SPARQL (SPARQL Protocol and RDF Query Language) query which is the de facto technique for querying Linked Data. This query is then executed on the target Linked Data resource from which the answers are extracted. Compared to the conventional QA approach, QALD provides many advantages. Firstly, as Linked Data contains structured knowledge in the form of triples (subject ↔ predicate ↔ object), this can be easily queried to produce answers with with reduced ambiguity compared to information extraction based techniques which involve error prone sub-processes such as, named entity extraction, relation extraction inter alia. Secondly, Linked Data contains massive amounts of interlinked information compared to a traditional knowledge base or a corpus [3].

Although, QALD offers great flexibility to the QA research, the answers extracted through QALD are still factoids. This is because a typical SPARQL query extracts only the requested information units (e.g., an object or a list of objects) from the structured Linked Data, very much similar to a SQL query executed on a relational database. However, numerous researches [22, 11, 21, 20] suggest that, in general, QA research should now extend the focus to the presentation aspects once the answer is extracted. The need for presentation is also influenced by recent developments in related domains such as Intelligent Personal Assistant (IPA) research (e.g., Apple Siri, Microsoft Cortana, Google Now) [13].

This paper introduces the concept of answer sentences as a presentation mechanism for QALD. An answer sentence is a linguistically complete natural language sentence which embeds the acquired answer, built by utilizing the linguistic structure of the source question. The technique combines syntactic as well as semantic approaches to generate the answer sentences which is also akin to how humans construct answer responses to questions. To the best of our knowledge, no previous study has investigated generating an answer sentence utilizing the source question's linguistic structure.

The rest of the paper is structured as follows. Section 2 discusses some of the related approaches. Due to the absence of approaches which closely resemble our objective, we provide a discussion on a broad context of answer presentation. Section 3 focuses on the methodological details which is followed by Section 4, which explains the experimental framework. Finally, in Section 5 we conclude the paper with an overview of the future work.

## 2 Related Work

As mentioned in Section 1, the theory behind the answer presentation described in this paper has not been experimented in earlier literature. However, some indirectly related work motivated our current approach.

Bosma [4] presents a summarization based approach for answer presentation. In this approach the sentences in the document where answer is appeared are extracted and then analysed for existence of rhetorical relations [16]. The resulting Rhetorical Structure (RS) is transformed to a weighted graph which is then used to select sentences that are more important to the context. Although the approach looks promising, it has several drawbacks. The key challenge here is that identification of rhetorical relations among individual sentences in unstructured text is a complex and difficult task. Bosma [16] assumes that rhetorical structure of the document is available from beforehand, however, in fact such resources are rare and inaccurate. One possible alternative would be to employ a model to extract rhetorical relations from the document. However, these are extensively error prone as sentences may link to other sentences in the document, not only via simple semantic relations, but also though more complex and hidden semantic relations which current models are incapable to identify. Bosma's model differs from our approach significantly. Our focus in this research is to utilize the source question as the basis for presenting the answer, while Bosma attempts to use pre-constructed sentences in the source document to present the answer.

Significant efforts to present answers in QA system is evident in the cooperative QA systems. A cooperative QA system is an advancement of a general QA system which tries to provide useful answers to the user even in situations where a specific answer is not to the user's question is not available. A widely used strategy in cooperative QA systems is the intensional answering where instead of retrieved answers (extensional answers) to a question, an answer is generated which represents the characteristics of the retrieved answers. Beneramara [2] mentions the five strategies of intensional answers: introducing higher level concepts, data reorganization, generalization, quantification, and correlation in the elements of the question. The generalization in the intensional answers is similar in context to our problem. The generalization looks at all the answer candidates and generates an answer which can better describe the answer set. Although Benamara [2] does not use the linguistic structure of the source question as in our approach, the generated answer can be presented as a sentence. For instance, Benamara [2] explains an example scenario for the question "which country can I visit without any visa formalities?". For this question a potential answer from a traditional QA system will generate a list of countries as the answer. However, a cooperative QA system with intensional answering capabilities will generate an answer in the form of "All the countries of the EEC except the UK and Norway". This type of a sentence which represents the generalized form of the extensional answer set humanizes the QA systems by providing a more natural answer. However, this paper focuses on generating a sentence which uses the same linguistic structure that question contains. This sort of answering mechanism conveys the message to the user that QA system properly understands the message and the language constructs that are embedded in the user's question.

Similar to the work carried out by Benemara [2], Moriceau [24] explains the numerical data integration in

cooperative answers by generating a sentence with integrated answer set. Moriceau uses the four relations (inclusion, equivalence, aggregation, and alternative) that are defined by Webber et al [35] to integrate answers. Although Moriceau [24] does not focus more on language generation process to generate the sentence with the integrated answer, an overview of the generation process is proposed.

Yu et al [36] present the centroid based summarization [32] for answer formulation in the MedQA system which concentrates on generating definitional answers for physicians. Since the summarization method relies on the centroid based approach, Yu et al [36] employ group-wise average and single pass clustering selectively to cluster sentences to support the summarization model. Although the MedQA is evaluated using human participants focusing on the usability of the entire system, the evaluation set is too small (12 questions) to derive important conclusions and generalize the finding of the research. Compared to the MedQA, our approach utilizes 52 questions and evaluated with multiple dimensions which will be described in Section 4. An inherent drawback to summarization based answer presentation is out-of-focus summaries that will be generated if the answer appears in a document which is not in the same context as the question. MedQA is also exposed to this drawback since it relies on the lexical similarity to remove redundant sentences instead of a combination of lexical and semantic features. Similar approaches to MedQA can also be seen in the work carried out by Demner-Fushman and Lin [6], which utilizes the MEDLINE sentence summarization approach to present the answer.

Vargas-Vera and Motta [34] report the implementation of AQUA QA system which uses a domain ontology to enhance the final answer. AQUA retrieves the answer related concepts from the domain ontology and then transforms these concepts to natural language. Since the ontology concepts are related to each other with standard relations set, a controlled natural language based approach can be employed for the transformation. A key challenge here is the identification of domain ontology to extract concepts. Although this is possible in closed domain QA, finding such ontology is extremely hard in open domain QA which deals with multiple domains.

We described a number of answer presentation mechanisms here which essentially focus on naturalizing the extracted answer. However, the approach we propose in this paper differs from these approaches where we concentrate on using the linguistic structure of the source question to present the answer as an answer sentence.

## 3 RealText$_{\mathbf{asg}}$: Answer Sentence Generation

This section focuses on describing the answer sentence generation system in detail. We start our discussion by introducing the system architecture and then proceed to explain individual modules in detail.

### 3.1 System Architecture

Figure 1 depicts the system architecture of the answer sentence generation. The architecture comprises of two phases; one which focuses on extracting patterns and the other which utilizes the extracted patterns to generate answer sentences. The pattern extraction phase extracts typed dependency subtree patterns from the development question dataset and preserved in a database after filtering the duplicates. In the database we classify the patterns with respect to two main question types: wh-interrogatives and the polar interrogatives. The answer sentence generation phase contain individual functions for each of the pattern recorded in the database and given a new question and answer pair, this phase can then generate an answer sentence by applying the correct pattern and embedding the answer. In addition, answer sentence generation phase further realizes the answer sentence targeting a better readability and accuracy level.

### 3.2 Question Classification

Natural language questions can be classified into two main clusters based on the interrogative types: wh-interrogatives and polar interrogatives [33,10]. In addition to these interrogative types, an imperative statement (e.g., "Tell me the birth date of Michael Jordan") can also be used to satisfy the purpose of a question. However, according to Materna [19], imperative statements cannot be generally considered as interrogatives. Hence the current research do not consider imperative statements.

A wh-interrogative is a question which contains a wh token (e.g., which, what, when, who). All wh-interrogatives except ones that are based on token "*why*" expect factoid answers. The wh-interrogatives with token "*why*" always require definitional answers. Since our study is focused only on QALD which targets factoid answers, we do not consider wh interrogatives based on token "*why*".

A polar interrogative is a question that can be answered with true or false value – conforming or denying the statement mentioned in the question. For instance, the question "*Did Socrates influence Aristotle?*" seeks

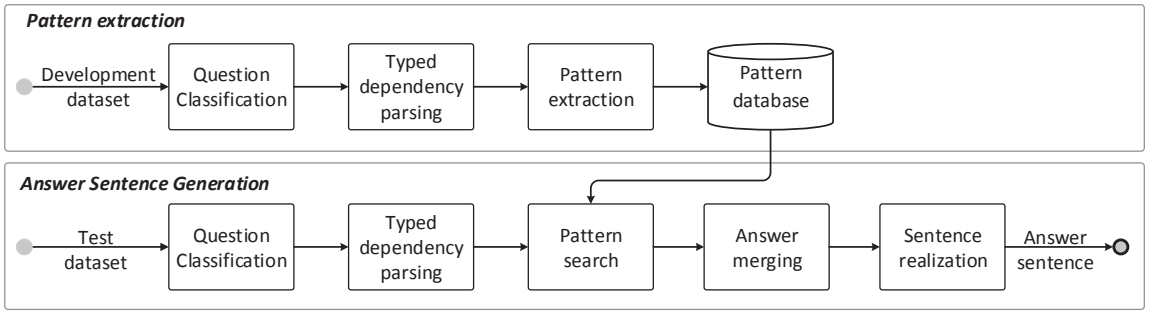



**Fig. 1** System architecture of the answer sentence generation

the truth value of the statement "*Socrates influenced Aristotle*".

Our preliminary analysis on the answer sentences showed that answer sentences for the two types of questions vary significantly. In essence, answering a wh-interrogative introduces a new entity or a property of a mentioned entity. This new information must be encoded when answering the wh-interrogative question with an answer sentence. For instance, the question "*When was the Statue of Liberty built?*" seeks the opening date property of the "*Statue of Liberty*" which is October 28, 1886. The answer sentence must embed this new information which will result in "*The Statue of Liberty was built on October 28, 1886*". On the other hand polar interrogatives need a negation or same statement being restated to form an answer sentence.

Since our study concentrates on answer presentation, we utilized both natural language question and the query for the question type classification. The query which is a formalized version of the natural language question is of course easy to process. However, it lacks some information such as existence of wh interrogative. Hence using both we implemented a hybrid rule based question type classifier.

The first step of the question classification is to identify the polar interrogatives. This is quite straightforward as polar interrogatives which seek truth value must always represent as an ASK SPARQL query, as shown in Listing 1 for a sample question. We parse the SPARQL query using Jena parser[1] and identify the type and if it is a ASK query then it is classified as polar interrogative.

[1] `https://jena.apache.org`

**Listing 1** ASK SPARQL query for the question "Was the Cuban Missile Crisis earlier than the Bay of Pigs Invasion?"

```
PREFIX dbo: <http://dbpedia.org/
    ontology/>
PREFIX res: <http://dbpedia.org/
    resource/>
ASK WHERE
{
res:Cuban_Missile_Crisis dbo:date ?x .
res:Bay_of_Pigs_Invasion dbo:date ?y .
FILTER (?x < ?y)
}
```

Once the classification of polar interrogatives is finished, the rest can be classified as wh-interrogatives. To further affirm the wh-interrogative classification, we Part-of-Speech (POS) tagged the natural language question to check whether it contains a *wh* token. Table 1 depicts the POS tags associated with wh-interrogatives. Table 1 also provides some examples on both question types for a wider comparison.

### 3.3 Dependency Subtree Pattern Extraction

In following sections we discuss the dependency parsing and what it meant by a dependency subtree. We start our discussion by first presenting the foundation of dependency parsing and some basic information as it is central to our approach of answer sentence generation.

#### *3.3.1 Dependency Grammar and Parsing: An Overview*

Dependency Grammar [25] introduces the concept of syntactic formation where individual tokens are linked through asymmetrical relations known as dependency relations. In essence, the dependency relation connects

**Table 1** Sample scenarios for interrogative types. The table list the associated POS tags, interrogative tokens, and some example scenarios from both question types.

| | Wh-interrogative | Polar interrogative |
|---|---|---|
| Interrogative tokens | What, When, Which, Where | Is, Are, Was, Were |
| POS tags | WP, WDT, WRB | VBP, VBD, VBZ |
| Question-1 (POS tagged) | WP(*Who*) VBD(*was*) DT(*the*) NN(*successor*) IN(*of*) NNP(*John*) NNP(*F.*) NNP(*Kennedy*) | VBD(*Was*) NNP(*Margaret*) NNP(*Thatcher*) DT(*a*) NN(*chemist*) |
| Answer | Lyndon B. Johnson | True |
| Answer sentence | The successor of John F. Kennedy was Lyndon B. Johnson | Margaret Thatcher was a chemist |
| Question-2 (POS tagged) | WRB(*How*) JJ(*many*) NNS(*students*) VBZ(*does*) DT(*the*) NNP(*Free*) NNP(*University*) IN(*in*) NNP(*Amsterdam*) VB(*have*) | VBD(*Was*) JJ(*Dutch*) NNP(*Schultz*) DT(*a*) NN(*Jew*) |
| Answer | 22730 | False |
| Answer sentence | The Free University in Amsterdam has 22730 students | Dutch Schultz was not a Jew |

two tokens, one which governs the relation (head) and the other which depends (dependent). As dependency grammar expects each token of the sentence to have a head, we insert an artificial root node which actually becomes the head of the sentence to support the theoretical and computational processing of dependency grammar. Fig. 2 shows an example question encoded with dependency grammar relations.

The dependency parsing is the process of identifying the dependency structure of a given sentence automatically. The dependency parsers represent two major camps; data driven parsing and the grammar based parsing. The data driven parsing can be further subdivided into two classes: transition based and graph based parsing. Similarly, context-free and constrained based parsing are two categories that grammar based parsing can be subdivided into. In this research, we utilized the Stanford Parser [17], a CFG grammar based parser utilizes universal typed dependencies which will be discussed in Section 3.3.2.

In our problem of dependency parsing, we denote a question $Q$ which is composed of tokens $q_0, q_1 \dots q_n$ where $q_o$ is the artificially inserted root node. Consequently, $R = \{r_1, r_2, \dots, r_m\}$ is a finite set of possible dependency relations types that link two tokens in the $Q$. We can define the dependency tree for question $Q$ as a directed tree $T_Q$ where $T_Q = (V, A)$. Here, $V$ is the spanning node set of $T_Q$ meaning that $V \subseteq \{q_o, q_1 \dots q_n\}$ and $A$ denotes the arcs where $A \subseteq V \times R \times V$. And importantly $T_Q$ originates from the $q_0$ satisfying the root property which infers that there cannot be a $q_i \in V$ such that $q_i \rightarrow q_0$.

### *3.3.2 Dependency Subtree Patterns*

A dependency subtree in our study can be defined formally as $T_{QS} = (V_x, A_x)$ where $A_x \subseteq V_i \times R \times V_j$ and $V_x = V_i \cup V_j$. The $V_i$ and $V_j$ can be defined as $V_i \subseteq \{q_i | (q_0, r, q_i) \in A\}$ and $V_j \subseteq \{q_j | q_j \in V \setminus V_i\}$ respectively. This formalism limits our dependency tree to a subtree which originates from the dependent of the artificial root node.

We then extract the patterns using the subtrees identified from dependency parsing. A pattern in our approach constitutes to the dependency relations appear in the subtree. We do not pay any attention on the actual tokens or their associated POS tags during the pattern extraction. This is because we only concentrate on the syntactic structure from the perspective of root and not the underlying word level features. Table 2 denotes an example set of dependency subtrees and patterns extracted from the original dependency trees of parsed questions. The extracted patterns represent a mere listing of relations. However, to generate

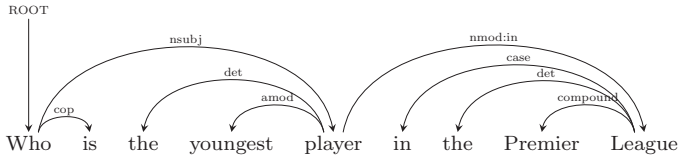
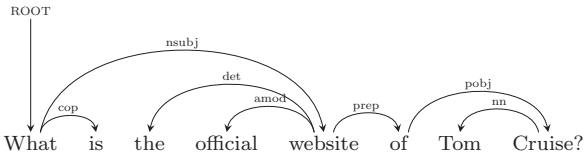
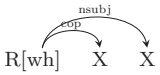
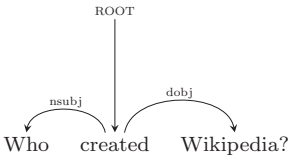
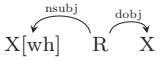
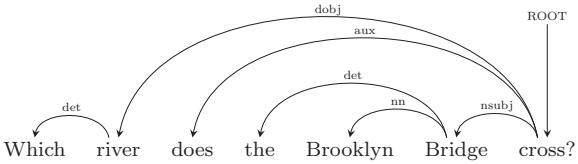
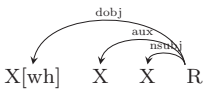
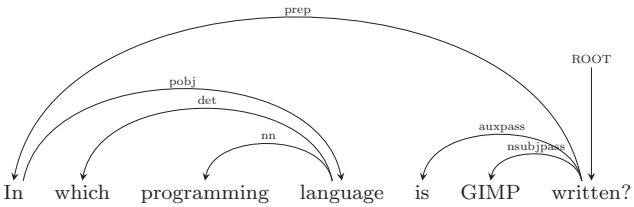
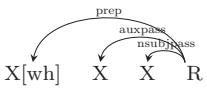



**Fig. 2** An example depicting dependency grammar relations between tokens in a question.

**Table 2** Examples of dependency subtrees extracted from parsed questions. The questions are taken from QALD-2 test dataset.

| Typed Dependency Tree | Typed Dependency Subtree | Dependency Pattern |
|---|---|---|
|   |   | *nsubj* ↔ *cop* ↔ *Root* |
|   |   | *nsubj* ↔ *Root* ↔ *dobj* |
|   |   | *nsubj* ↔ *aux* + *Root* ↔ *dobj* |
|   |   | *nsubjpass* ↔ *auxpass* + *Root* ↔ *prep* |

a sentence utilizing these relations, the order of appearance must be declared. The final column in the Table 2 shows the ordered relations which can be used as the finalized pattern to generate an answer sentence.

More importantly we use the universal typed dependencies [18] in our framework. The universal typed dependencies defines a taxonomy of grammatical relations which can be used across languages. This solves the key challenge in dependency parsing by allowing them to adopt for a number of languages to identify the syntactic structure. Further work on universal typed dependencies are still in progress which include mapping existing dependency schemes to this universal taxonomy [18]. A main reason that motivated us to employ universal typed dependencies is the opportunity to consider our current approach in a different language in future. However, we also support dependency schemes which do not comply with universal schema. This is achieved technically by mapping universal typed dependencies to a framework specific typology for easy configuration.

The extracted patterns are preserved in a database and are used to generate the answer sentences. The process of searching and applying these patterns is explained in the following section.

### 3.4 Searching and applying a pattern

When a new question and answer pair is provided to generate the answer sentence, the question is first dependency parsed and relations are extracted through the root level subtree. However, we have no prior knowledge on the ordering of the relations. Therefore, we search the pattern database without considering the order of the relations and consider only the possible existence. For instance, a possible pattern $\langle nsubj,\ cop,\ dobj \rangle$ is considered as a matching pattern for the newly derived set ($dobj$, $cop$, $nsubj$) which is unordered. At this level of processing we have a clear idea on how the new answer sentence should be syntactically structured based on the source question, but has no idea on the content.

The pattern application stage looks at what content should be included in the answer sentence which will be taken from the source question. The content is derived by considering all the associated tokens in the subtrees. These tokens are now transformed into individual phrases following the same order that appear in the source question. However, we do not transform the phrase that contains the wh-token to a textual phase at this stage. This is mainly to support answer embedding process which is explained in Section 3.5.

Then we can order the appearance of these phrases to form an answer sentence. This is carried out by consulting the order of relations in the pattern that is selected for that particular question. Some example scenarios for phrase extraction based on the dependency tree and their ordering are shown in Table 3.

### 3.5 Embedding the Answer

The answer needs to be embedded once the pattern is applied on the new question. The process of embedding the answer depends on the question type as described in Section 3.2.

In fact polar interrogative do not need answer merging, but a modification of the polar token is required if the statement is false, otherwise the question can be restructured to form the answer sentence. In essence, we identify the polar token in the dependency parsed question and then negate it (e.g., is ⇒ is not) if the answer is false and in case the answer is true we do not alter the polar token except the restructuring.

For wh-interrogatives, we consult the dependency subtree with wh-token which is kept as it is without transforming to a phrase (see Section 3.4). The answer embedding is carried out through predefined set of rules considering the linguistic structure of the dependency subtree with wh-token. The rules with examples are shown in Table 4.

In addition to embedding the answer, this module also formulates the answer to make it more natural and accurate. First if the module determines that the answer appears at the beginning of the sentence, then the numerical answers are verbalized into literal forms (e.g., 224 ⇒ two hundred twenty four). This is carried out through a rule based number verbalizer which is implemented to work on vast range of numbers.

The answers which require measurement units are then associated with short name of the measurement unit. To identify measurement unit we exploit the SPARQL query. The framework first parses the SPARQL query, extracts the basic graph patterns, and embeds triples mentioned in the query. Listings 2 and 3 show a SPARQL query and a resulting SPARQL algebraic definition for basic graph patterns. Although the example depicts a scenario with a one triple, a more complex SPARQL query can result in multiple basic graph patterns each having multiple triples. In the next step we screen through the triples and find out the triple which contain the queried variable (e.g., *?num* in the example shown in Listing 2). The predicate in this triple is the queried predicate and if this predicate is identified as one which needs a measurement unit, then the answer is associated with the appropriate measurement unit. However, to identify this we need a database which contains records of predicates which need measurement units and some basic information on these measurement units (i.e., long name and the short name of the measurement unit). This database is created as a preliminary work of this project and currently contain 34 predicates with their associated measurement units information. Sample set of records from this database is shown in Table 5.

**Listing 2** A SPARQL query for the question "How tall is Michael Jordan?"

```
PREFIX dbo: <http://dbpedia.org/
    ontology/>
PREFIX res: <http://dbpedia.org/
    resource/>
SELECT ?num WHERE
{
        res:Michael_Jordan dbo:height ?
            num .
}
```

**Listing 3** Algebraic expression of the SPARQL shown in Listing 2

```
(project (?num)
(bgp
```

**Table 3** Extracting phrases and ordering them based on the selected pattern

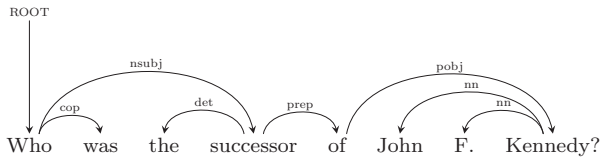
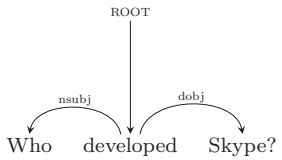
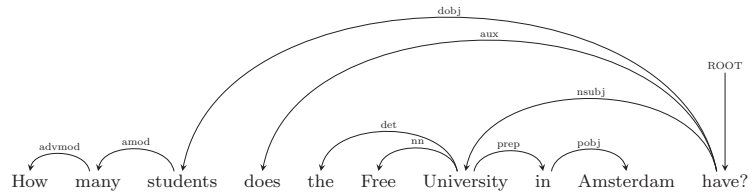
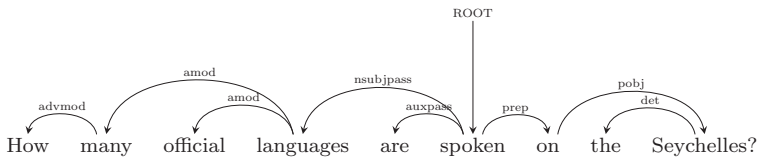
| Parsed Question | Selected Pattern | Extracted Phrases |
|---|---|---|
|  | *nsubj* ↔ *cop* ↔ *Root* | who, was, the successor of John F. Kennedy |
|  | *nsubj* ↔ *Root* ↔ *dobj* | who, developed, Skype |
|  | *nsubj* ↔ *aux* + *Root* ↔ *dobj* | how may students, does, the Free University in Amsterdam, have |
|  | *nsubjpass* ↔ *auxpass* + *Root* ↔ *prep* | how many official languages, are, spoken, on the Seychelles |

**Table 4** Answer embedding for wh-interrogatives

| Wh token | Merging rule | Example phrase | Merged answer |
|---|---|---|---|
| Which | Preposition + Factoid Answer | in which city | in Auckland |
| What | Preposition + Factoid Answer | for what company | for Google |
| Whom | Preposition + Factoid Answer | for whom | for Steve Jobs |
| How | Verbalized Answer (once/ twice/..) | how often | thrice |
| | Factoid Answer+ Rest of the phrase | how many cars | 3 cars/ Three cars |
| When | New preposition + Factoid Answer | when | on 15 March, 2015 |
| Where | New preposition + Factoid Answer | where | in Melbourne |

```
(triple <http://dbpedia.org/resource/
    Michael_Jordan> <http://dbpedia.org/
    ontology/height> ?num)))
```

### 3.6 Further Realization

The further realization concentrates on improving the readability of the generated answer sentences. The structure of the sentence and the grammatical formalism are all finalized at this stage of processing. However, there still exist periphrastic tense embedded in the generated answer sentence (e.g., Did Socrates influence Aristotle? ⇒ Socrates *did influence* Aristotle). In many scenarios to form a question from a statement, a periphrastic tense must be used instead of an mere inflection. This periphrastic tense is included in the answer sentence by default since our framework is based on using the same linguistic structure of the source question to form the answer sentence.

**Table 5** Sample set of record from the measurement unit database

| Predicate | Ontology URI | Unit Long name | Unit Short name |
|---|---|---|---|
| areaTotal | http://dbpedia.org/ontology/areaTotal | square meter | $m^2$ |
| netIncome | http://dbpedia.org/ontology/netIncome | US Dollars | USD |
| length | http://dbpedia.org/ontology/length | meter | m |
| discharge | http://dbpedia.org/ontology/discharge | cubic meter | $m^3$ |
| runtime | http://dbpedia.org/ontology/runtime | seconds | s |

The further realization module realizes the periphrastic tense using a verb information database which is built based on the VerbNet [12]. The VerbNet is used only to get the required coverage of the English verbs and beyond that it does not provide the inflections of the verbs except the verb frames. We used the both SimpleNLG [9] and DictService [1] to get the required verb inflections. Each verb in base form is associated with the past tense, past particle form, progressive form, and the third person singular form. The database has further applications beyond this particular usage and some of them are discussed in our related projects [27–30]. Table 6 lists a sample set of the records from this database which currently contains 3773 records. We utilize this database to further realize the sentences to a more natural form transforming all periphrastic tenses to inflectional forms (e.g., Socrates *did influence* Aristotle ⇒ Socrates *influenced* Aristotle).

## 4 Evaluation Settings and Results

The evaluation of the framework was three folds. The first evaluation phase focused on assessing the linguistic accuracy of the answer sentences. We then performed a human evaluation to rate answer sentence on both readability and accuracy. The third evaluation phase focused on a feasibility analysis of using automatic metrics to evaluate the generated answer sentences against the human provided reference answer sentences. The following sections provide detailed information on the datasets and evaluations.

### 4.1 Datasets

For the evaluation we utilized the QALD-2 train and test datasets which contain both factoid and list questions. In this research we do not focus on list based questions which request long list of information from the Linked Data resource. This is mainly due to two reasons; firstly list based questions are mostly formed as imperative constructs (e.g., List all cities in Germany.), and secondly it is not meaningful to generate an answer sentence from a long list which may contain excessive number of items (e.g., 2059 different cities in Germany). Although latter can be accomplished by shortening the list (e.g., Berlin, Munich, Hamburg, Frankfurt, etc.), such representation will eventually lead to an information loss from the user's perspective as all requested information is not presented in the answer.

We used the QALD-2 train dataset as our development dataset from which we extracted the dependency patterns while test dataset is used to test the framework by applying the extracted patterns. The development and test datasets comprised of 50 and 52 questions respectively.

### 4.2 Linguistic accuracy analysis

The pattern extraction process resulted in 25 patterns out of which 18 are wh-interrogative patterns and 7 are polar interrogative patterns. Although it is possible to further categorize the patterns based on some features (e.g., wh token, relation types), we do not carry out such clustering as it is not related to the objective of this research.

The framework generated 41 linguistically correct answer sentences for the test dataset which comprised of 52 questions reporting a 78.84% linguistic accuracy level. The framework failed to generate answer sentences for 11 questions (five wh-interrogatives and one polar interrogative question). The main reason of this failure was that generated patterns did not cover the test scenarios (for 10 questions), and the second reason was the errors in dependency parsed question.

We then analysed the coverage provided by the extracted patterns in the test dataset. Fig. 3 and 7 depict the coverage of top 10 patterns and a list of top-5 patterns respectively. According to Fig. 3, the top-10 patterns were able to cover 73.07% while only the top-4 patterns covered 53.84% of the testing scenarios.

**Table 6** Sample set of records from the verb information database built based on the VerbNet.

| Base form | Past tense | Past participle | Progressive form | Third person singular | Frame types |
|---|---|---|---|---|---|
| abridge | abridged | abridged | abridging | abridges | NP.patient V, ... |
| accept | accepted | accepted | accepting | accepts | NP V NP, ... |
| activate | activated | activated | activating | activates | NP.patient V, ... |
| advertise | advertised | advertised | advertises | advertises | NP V PP.location, ... |

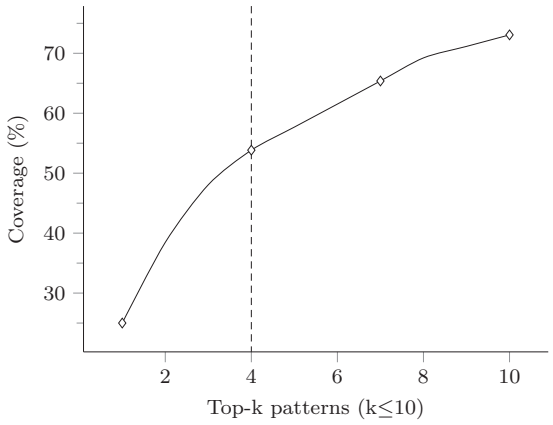



**Fig. 3** Coverage of the extracted patterns in test dataset

| |
|---|
| $nsubj \leftrightarrow cop \leftrightarrow Root[wh]$ |
| $nsubj[wh] \leftrightarrow Root \leftrightarrow dobj$ |
| $nsubj \leftrightarrow Root + aux \leftrightarrow dobj[wh]$ |
| $nsubjass[wh] \leftrightarrow Root \leftrightarrow auxpass \leftrightarrow prep$ |
| $nsubj \leftrightarrow Root \leftrightarrow dep[wh]$ |

**Table 7** Top-5 Patterns

### 4.3 Human rating based evaluation

The human rating based evaluation focused on assessing the readability and the accuracy of the generated answer sentences. For this task we hired three post-graduate students who have shown a good level of proficiency in English (all of them have done IELTS and had scored greater than 6.0). Prior to the evaluation, we carried out a pilot run to confirm that the participants ratings are meaningful using sample set of questions and answer sentences which are not included in the test set.

The participants were provided the 41 questions for which the system was able to generate answer sentences. They were asked to rate each answer sentence for readability and accuracy based on a 5-point Likert scales as shown in Fig. 4. Each Likert item was coded with both string representing the rating judgement and the corresponding numerical value where rating value "5" (very good) was the highest and the value "1" (very poor) was the lowest.

The evaluation exercise resulted in 123 complete responses. The inter-rater agreements within participants calculated using Cronbach's Alpha were 0.842 and 0.771 for accuracy and readability respectively. The higher

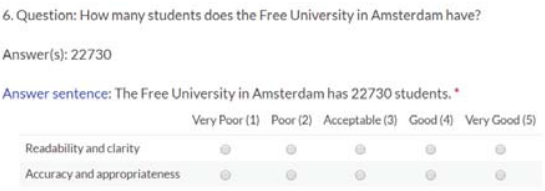



**Fig. 4** Screenshot of the answer sentence evaluation survey

level Cronbach's Alpha values show that all the raters completed the task based on a common ground. The analysis results of this task is summarised in Table 8. According to Fig. 8, it is clear that all participants have rated the answer sentences based on a common agreement as removal of any participant cannot increase the Cronbach's alpha for accuracy or readability.

Fig. 5 depicts the weighted average of rating results of both accuracy and the readability for the 41 questions. According to Fig. 5, for 37 answer sentences (represents 90.24% of the generated answer sentences), participants have rated the accuracy score as Likert value of 5 (very good). The readability ratings show that 31 answer sentences (represents 75.60% generated answer sentences) are rated with Likert value of 5 by all three

**Table 8** Summary of the statistics related to inter-rater agreement in evaluation

| Participant | Accuracy | | | Readability | | |
|---|---|---|---|---|---|---|
| | Cronbach's Alpha | Item-total correlation | Alpha if Item Deleted | Cronbach's Alpha | Item-total correlation | Alpha if Item Deleted |
| P1 | 0.842 | 0.707 | 0.781 | 0.771 | 0.559 | 0.745 |
| P2 | | 0.707 | 0.781 | | 0.689 | 0.593 |
| P3 | | 0.707 | 0.781 | | 0.584 | 0.717 |

participants. The results show that majority of the generated answer sentences are not only accurate, but also readable as well.

In addition to this evaluation, we also carried out further analysis on the data to find any potential existence of relationship between the accuracy and readability. A two-tailed Spearman Correlation test resulted in 0.612 correlation coefficient ($p$<0.001) between the readability and accuracy of the answer sentences. This revealed that although human ratings for readability and accuracy overlap for more than 85% of the cases, there does not exist a strong positive correlation between the two criteria.

### 4.4 Automatic evaluation

Our objective in this evaluation phase was to see the viability of using machine translation based automatic metrics for answer sentence evaluation. We first evaluated the answer sentences using two automatic metrics, Meteor [7] and BLEU [26], and then analysed whether the results can correlate with human ratings provided in Section 4.3.

The Meteor uses four modules to align a human reference sentence and a machine generated sentence to assign the score. The four Meteor modules are, exact matcher, stem matcher (uses the Snowball Stemmer [31]), synonym matcher (based on WordNet [23] synonyms), and the paraphrase matcher. This metric is more focused on sentence based evaluation than the BLEU metric which works on n-gram statistics.

The BLEU is a measure based on unigram co-occurrence statistics. Although BLEU is more focused on corpus level analysis, however, we used it with smoothing techniques to work with sentence level analysis as described by Chen and Cherry [5]. Chen and Cherry [5] introduce four traditional smoothing techniques and three new techniques to use with BLEU. A brief description of these smoothing techniques are provided in Table 9.

The availability of reference sentences provided by human participants is the main requirement to use the automatic metrics like BLEU and Meteor. We provided the question and answer pairs to participants and asked them to come up with answer sentences. Two examples were also shown to participants to guide them how exactly it should be written. In addition, the necessary measurement unit was shown together with the answer. The participants provided answer sentences for a random sub-sample of 32 questions.

Fig. 6 shows the evaluation results for Meteor and BLEU with four widely used smoothing techniques which corresponds to first four techniques described in Table 9. According to the results, in 9 scenarios Meteor and BLEU smoothing techniques except S4 have reported score of 1 which is considered as perfect matching of human and system answer sentences. In all the scenarios S4 smoothing technique has reported the lowest matching score. More importantly this technique has reported scores below 1.0 for the scenarios where all other metrics have reported the perfect matching where human and system answer sentences are exactly similar in content.

Fig. 7 shows the evaluation results for Meteor and BLEU with smoothing techniques proposed by Chen and Cherry [5]. These results also report the score as 1.0 for the exact similar answer sentences as noticed in Fig. 6. However, there are number of differences in the reported values. For example, the neither Meteor nor BLEU have reported a score above 0.56 for Q-2, however, smoothing technique BLEU-S7 has reported a score of 0.97. We carried out a correlation analysis to analyse the differences that exist within these different smoothing techniques.

Table 10 reports the result of correlation analysis performed within automatic metrics. The results show that BLEU-S7 has reported low correlation with BLEU-S1 and BLEU-S6 ($p$<0.05). In addition, compared to other metrics, BLEU-S7 has shown low correlation coefficients. In general the rest of the smoothing techniques correlate with others in an acceptable level, for instance,

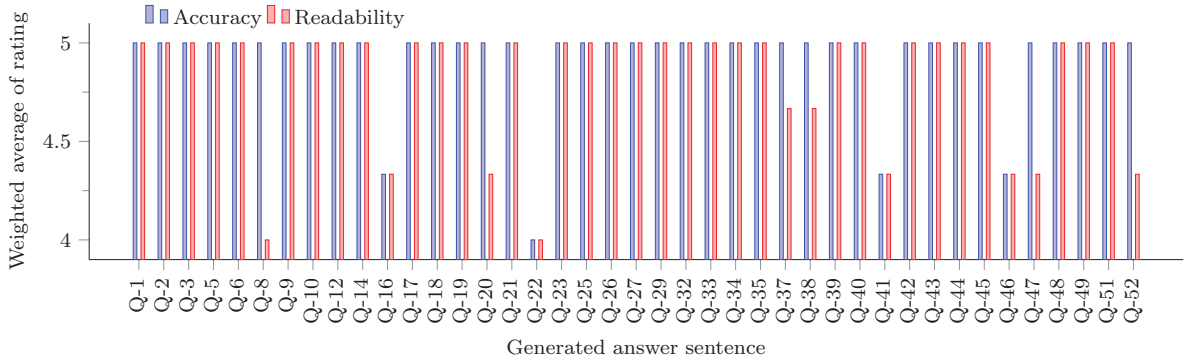



**Fig. 5** Weighted average of rating values for 41 questions (we have excluded 11 questions from the test set of 52 questions as our systems was unable to generate answer sentences for those questions)

**Table 9** A brief introduction to smoothing algorithms used with BLEU. The last three smoothing techniques are first proposed by Chen and Cherry [5]. These new ones are mostly formed by modifying the traditional ones shown in S1 to S4.

| Technique | Description |
|---|---|
| BLEU-S1 | This technique uses small positive value if the number of matched n-grams is 0. |
| BLEU-S2 | Details on this smoothing algorithm can be found in Lin and Och [14]. It is based on adding 1 to the matched n-grams count. |
| BLEU-S3 | This smoothing technique assigns geometric sequence to n-grams with 0 matches. The algorithm can be defined as below.<br>`invcnt = 1`<br>`for n in 1 to N`<br>`  if mn = 0`<br>`  invcnt = invcnt * 2`<br>`  mn' = 1/invcnt`<br>`  endif`<br>`endfor` |
| BLEU-S4 | Details on this technique can be found in Gao and He [8]. |
| BLEU-S5 | This technique is a modification of the BLEU-S3 proposed by Chen and Cherry [5]. It changes the line 4 of the BLEU-S3 algorithm to $invcnt = invcnt \times \frac{K}{ln(len(T))}$, where $K$ is set empirically. |
| BLEU-S6 | This is another novel method introduced by Chen and Cherry [5]. It is based on that matched counts for similar n gram sizes should be similar. |
| BLEU-S7 | This is the third novel smoothing technique introduced by Chen and Cherry [5]. In essence this technique combines the previous two techniques (BLEU-S5 and BLEU-S6). |

BLEU-S1 and BLEU-S6 show a strong correlation with
a correlation coefficient of 0.998 ($p<0.01$).

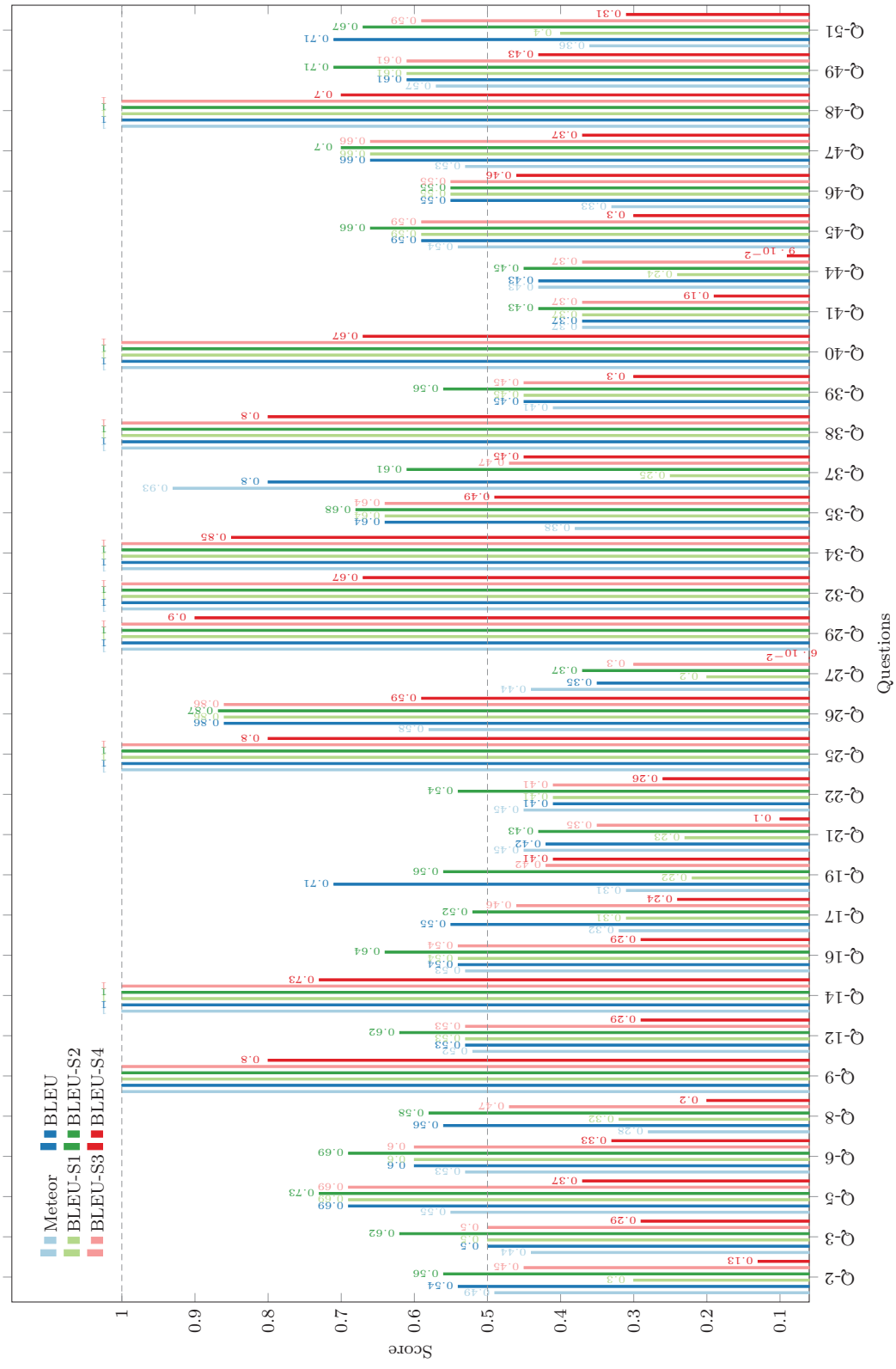



**Fig. 6** Automatic metric based evaluation results for Meteor and BLUE. BLEU is used with four traditional smoothing techniques.

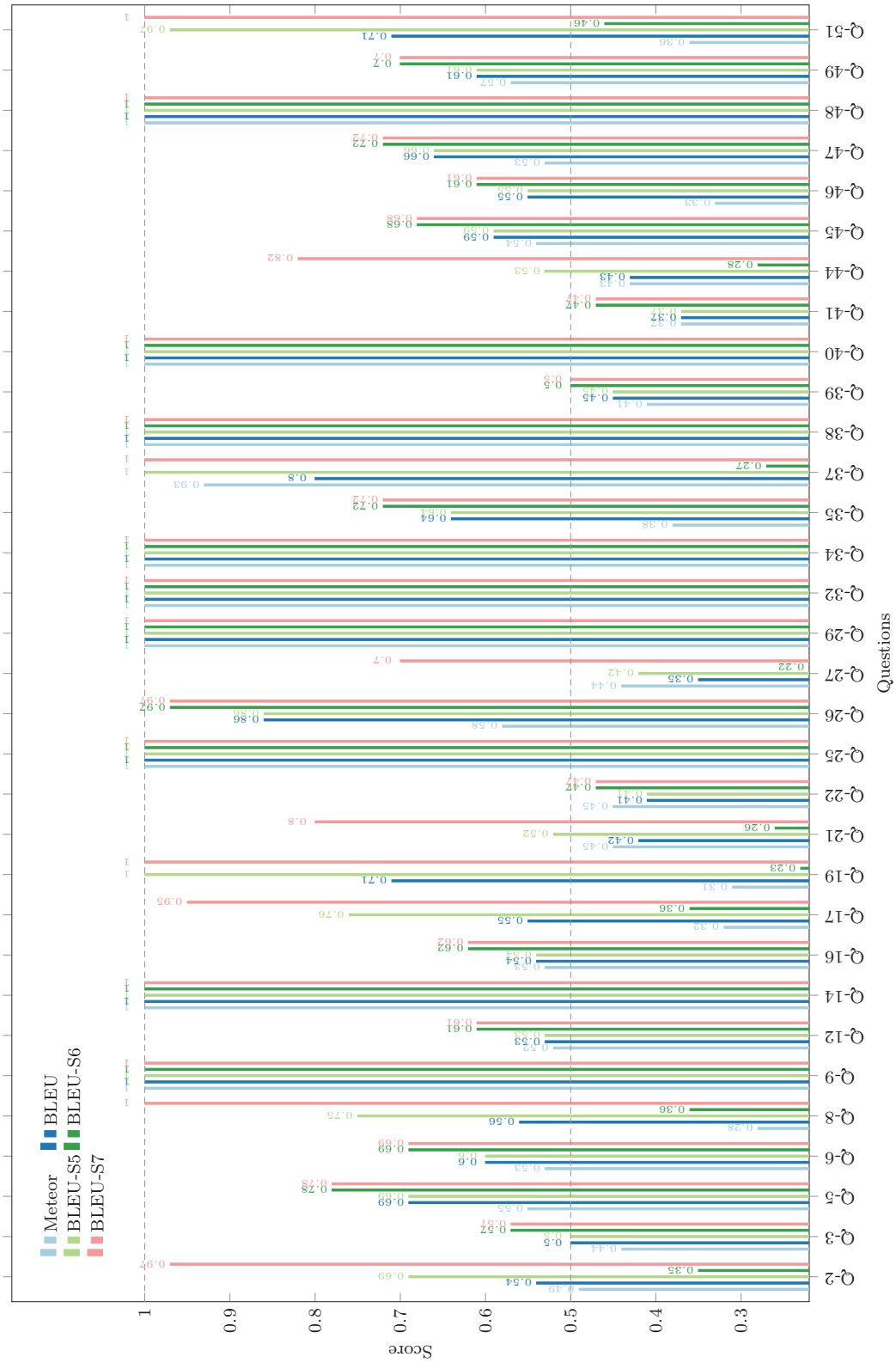



**Fig. 7** Automatic metric based evaluation for BLEU smoothing techniques proposed by Chen and Cherry [5]. Meteor and original BLEU are also reported for comparison.

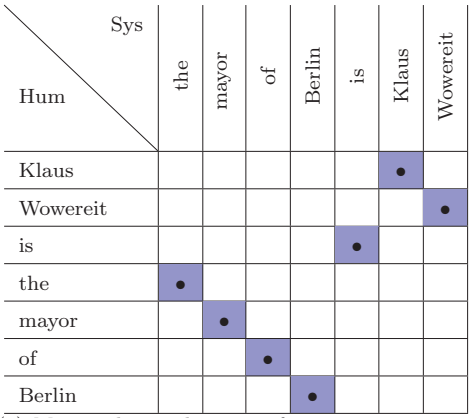
Meteor is considered for sentence level evaluation which essentially uses the alignments. We ran a visual alignment phase to further investigate the Meteor alignments. Fig. 8 shows the Meteor direct alignments where system and human answer sentences contain the same set of tokens, however, in different locations. In both Figs. 8(a) and 8(b), system generated answer sentence places the factoid answer at the end of the sentence, while human generated answer sentence places it at the beginning of the answer sentence. The framework assigns the factoid answer to the wh-token, and in both scenarios wh-token phrases are not the nominal subjects of the sentences. Since the framework builds the answer sentence based on Subject-Verb-Object (SVO) concept prioritizing the nominal subject, the factoid answer is placed at the end of the sentence. However, if the framework identifies the wh-token as the nominal subject, then in such cases factoid answer will be prioritized.

Fig. 9 depicts a scenario where Meteor paraphrase alignment is applied. The phrase "*married to*" and "*the husband of*" are identified as phrases having the same meaning and aligned accordingly. Although such alignment is very important, appropriateness of automatic metrics cannot be decided without investigating the correlation between the human ratings and automatic metric values.

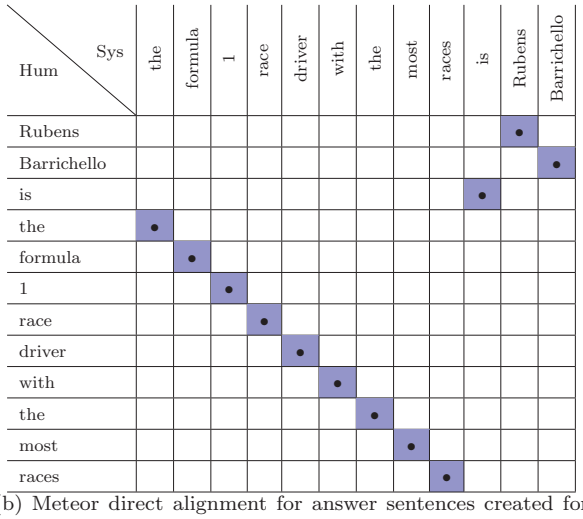
Since the automatic metrics in answer sentence evaluation is carried out as a feasibility study, an important phase was to measure the correlation between the Meteor and BLEU values with human ratings. Table 11 shows the results of correlation analysis between human ratings and automatic metric values. These results show that none of the automatic metrics has strong correlation with human provided ratings for accuracy or readability. The highest correlation reported was between BLEU-S7 and human provided accuracy ratings. However, its value 0.537 does not reflect a strong correlation. On the other hand, correlation between readability ratings and automatic metrics was very low and none of them reports a value with an acceptable significant level.

## 4.5 Discussion

This section described the three evaluations phases that we employed, and in the third evaluation phase we focused only on the feasibility of using automatic metrics. The linguistic accuracy evaluation confirmed that the framework can generate accurate answer sentences for more than 70% of the testing dataset. Furthermore, the analysis on pattern coverage revealed that the top-4




(a) Meteor direct alignment for answer sentences created for "Who is the mayor of Berlin?"




(b) Meteor direct alignment for answer sentences created for "Who is the Formula 1 race driver with the most races?"

**Fig. 8** Visualizations of Meteor direct alignments

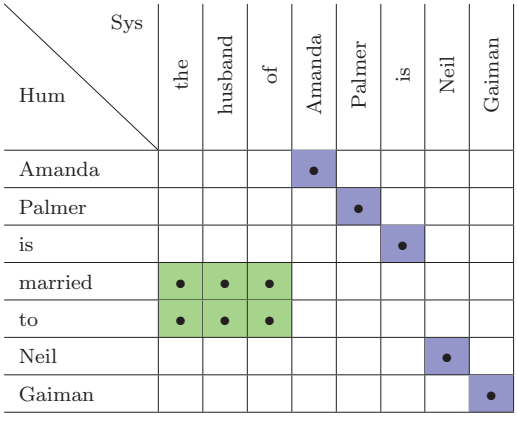



**Fig. 9** Meteor paraphrase alignment for the question "Who is the husband of Amanda Palmer?"

**Table 10** Inter-metric correlations for Meteor and BLEU variations. The ones that are marked with an asterisk (*) are has p<0.05 and the rest has p<0.01.

| | Meteor | BLEU-R | BLEU-S1 | BLEU-S2 | BLEU-S3 | BLEU-S4 | BLEU-S5 | BLEU-S6 | BLEU-S7 |
|---|---|---|---|---|---|---|---|---|---|
| Meteor | 1.000 | 0.730 | 0.800 | 0.834 | 0.793 | 0.728 | 0.599 | 0.800 | 0.471 |
| BLEU-R | 0.730 | 1.000 | 0.789 | 0.907 | 0.902 | 0.918 | 0.944 | 0.774 | 0.810 |
| BLEU-S1 | 0.800 | 0.789 | 1.000 | 0.937 | 0.964 | 0.861 | 0.579 | 0.998 | 0.373* |
| BLEU-S2 | 0.834 | 0.907 | 0.937 | 1.000 | 0.979 | 0.883 | 0.750 | 0.931 | 0.583 |
| BLEU-S3 | 0.793 | 0.902 | 0.964 | 0.979 | 1.000 | 0.904 | 0.738 | 0.955 | 0.551 |
| BLEU-S4 | 0.728 | 0.918 | 0.861 | 0.883 | 0.904 | 1.000 | 0.788 | 0.846 | 0.585 |
| BLEU-S5 | 0.599 | 0.944 | 0.579 | 0.750 | 0.738 | 0.788 | 1.000 | 0.563 | 0.925 |
| BLEU-S6 | 0.800 | 0.774 | 0.998 | 0.931 | 0.955 | 0.846 | 0.563 | 1.000 | 0.358* |
| BLEU-S7 | 0.471 | 0.810 | 0.373* | 0.583 | 0.551 | 0.585 | 0.925 | 0.358* | 1.000 |

**Table 11** Correlation analysis with automatic metrics with human ratings. The ones that are marked with an asterisk (*) and two asterisks (**) have significant levels of p<0.05 and the rest has p<0.01 respectively.

| Metric | Meteor | BLEU | BLEU-S1 | BLEU-S2 | BLEU-S3 | BLEU-S4 | BLEU-S5 | BLEU-S6 | BLEU-S7 |
|---|---|---|---|---|---|---|---|---|---|
| Correlation (Accuracy) | 0.271 | 0.418* | 0.157 | 0.369* | 0.300 | 0.229 | 0.464** | 0.153 | 0.537** |
| Correlation (Readability) | 0.239 | 0.245 | 0.169 | 0.272 | 0.225 | 0.232 | 0.181 | 0.178 | 0.245 |

patterns can cover more than 50% of the testing scenarios which shows that these patterns are highly representative. This confirms that using dependency subtree patterns is a good starting point for the answer sentence generation. The human evaluation focused on both the accuracy and readability of the generated answer sentences. More than 90% of the generated answer sentences were given highest rating for their accuracy and none of the generated answer sentences were rated below the score of 4. Furthermore, more than 75% of the answer sentences were given highest rating for the readability and again none of them were rated below the score of 4. This confirms that the proposed framework can generate high quality answer sentences which are both accurate as well as readable.

The automatic metric based evaluation used the Meteor and BLEU metrics. We carried out a visual alignment phase based on Meteor in addition to the general evaluation. This visual alignment phase revealed how Meteor aligns answer sentences considering the position of the tokens as well as considering the paraphrases. The BLEU metric is used under 8 settings where original BLEU metric and 7 smoothing techniques are applied. The inter-metric correlation showed that the correlation among these different metrics varies significantly. We then analysed the correlation between the automatic metrics and the human ratings. This showed that none of the automatic metrics can form a strong correlation with the human ratings for both readability and accuracy of the answer sentences. The results revealed that human evaluation is still the dominant and most trustworthy method for evaluation of language generation tasks.

## 5 Conclusion and Future Work

This paper presented the $\text{RealText}_{\text{asg}}$ answer sentence generation framework for QALD. The objective of the framework was to generate an answer sentence utilizing the source question linguistic structure while embedding the answer within. Since QALD is not enabled with answer presentation mechanism and there is no opportunity for summarization based presenta-

tion mechanism as in traditional QA systems, our proposed approach can be utilized to generate a more natural answer akin to human answer. The evaluation of the framework shows that it can generate accurate and readable answer sentences as evaluated by human users. We further extended our evaluation strategy to analyse the feasibility of the automatic metrics in answer sentence generation. This phase utilized BLEU and Meteor which have shown high correlation with human judgements in machine translation tasks. However, none of them were able to correlate with the human ratings in the answer sentence generation task. This is mainly due to the language variety where users can come up with different answer sentences for the same question.

In future, we plan to improve the $\text{RealText}_{\text{asg}}$ to generate multiple answer sentences for the same question utilizing the source question linguistic structure, the exact answer and additional contextual information. This will require us to come up with different strategies so that the dependency subtree can be used to generate sentences by modifying its structure and introducing new information. Furthermore, the core of the approach will be applied to traditional QA systems which do not use Linked Data as information source. This will require us to classify questions without analysing SPARQL queries and identify relevant metadata related to the answers automatically. In addition to the development tasks, the framework will be further evaluated with higher number of participants and further investigations will be carried out to identify automatic evaluation strategies that can result higher correlation with human ratings.

**Acknowledgements** The research reported in this paper is a part of a research funded by the Auckland University of Technology.

## References

1. Adrian O'Neill: DictService: Word Dictionary Web Service (2011)
2. Benamara, F.: Generating Intensional Answers in Intelligent Question Answering Systems. Natural Language Generation pp. 11–20 (2004). URL http://www.springerlink.com/content/ea4qdw16v6a6jbg0http://link.springer.com/chapter/10.1007/978-3-540-27823-8{_}2
3. Bizer, C.: The emerging web of linked data. Intelligent Systems, IEEE (2009). URL http://ieeexplore.ieee.org/xpls/abs{_}all.jsp?arnumber=5286174
4. Bosma, W.: Extending answers using discourse structure. In: Recent Advances in Natural Language Processing. Association for Computational Linguistics, Borovets, Bulgaria (2005). URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.216.4051
5. Chen, B., Cherry, C., Canada, C.: A Systematic Comparison of Smoothing Techniques for Sentence-Level BLEU Boxing Chen and Colin Cherry. Association for Computational Linguistics (ACL) (2), 362–367 (2014)
6. Demner-Fushman, D., Lin, J.: Answer extraction, semantic clustering, and extractive summarization for clinical question answering. In: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the ACL - ACL '06, pp. 841–848. Association for Computational Linguistics, Morristown, NJ, USA (2006). DOI 10.3115/1220175.1220281. URL http://dl.acm.org/citation.cfm?id=1220175.1220281
7. Denkowski, M., Lavie, A.: Meteor 1.3: Automatic metric for reliable optimization and evaluation of machine translation systems. . . . Workshop on Statistical Machine Translation pp. 85–91 (2011). URL http://dl.acm.org/citation.cfm?id=2132969
8. Gao, J., He, X.: Training MRF-Based Phrase Translation Models using Gradient Ascent. In: Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Association for Computational Linguistics, Atlanta (2013)
9. Gatt, A., Reiter, E.: SimpleNLG: a realisation engine for practical applications. In: Twelth European Workshop on Natural Language Generation, pp. 90–93. Association for Computational Linguistics, Athens, Greece (2009). URL http://dl.acm.org/citation.cfm?id=1610195.1610208
10. Ginzburg, J., Sag, I.A.: Interrogative Investigations. Stanford CSLI Publications (2000)
11. Hirschman, L., Gaizauskas, R.: Natural language question answering: the view from here. Natural Language Engineering **7**(04), 275–300 (2001). DOI 10.1017/S1351324901002807. URL http://dl.acm.org/citation.cfm?id=973890.973891
12. Kipper, K., Korhonen, A., Ryant, N., Palmer, M.: A large-scale classification of English verbs. Language Resources and Evaluation **42**(1), 21–40 (2008). DOI 10.1007/s10579-007-9048-2
13. Konstantinova, N., Orasan, C.: Interactive Question Answering. Emerging Applications of Natural Language Processing: Concepts and New Research pp. 149 —-169 (2013). DOI 10.4018/978-1-4666-2169-5.ch007. URL http://services.igi-global.com/resolvedoi/resolve.aspx?doi=10.4018/978-1-4666-2169-5
14. Lin, C.Y.: ROUGE: a Package for Automatic Evaluation of Summaries. In: Workshop on Text Summarization Branches Out. Association for Computational Linguistics, Barcelona, Spain (2004)
15. Lopeza, V., Urenb, V., Sabouc, M., Mottab, E.: Is Question Answering fit for the Semantic Web ?: a Survey . (2010)
16. Mann, W.C., Thompson, S.A.: Rhetorical Structure Theory: Toward a functional theory of text organization. Text **8**(3), 243–281 (1988)
17. Manning, C., Bauer, J., Finkel, J., Bethard, S.J., McClosky, D.: The Stanford CoreNLP Natural Language Processing Toolkit. In: The 52nd Annual Meeting of the Association for Computational Linguistics. Association for Computational Linguistics, Baltimore (2014)
18. de Marneffe, M.C., Dozat, T., Silveira, N., Haverinen, K., Ginter, F., Nivre, J., Manning, C.D.: Universal Stanford Dependencies: A cross-linguistic typology. In: 9th International Conference on Language Resources and Evaluation (LREC'14), pp. 4585–4592 (2014). URL papers3://publication/uuid/D4B7BB39-4FFB-4AA6-B21E-701A91F27739
19. Materna, P.: Question-like and non-question-like imperative sentences. Linguistics and Philosophy **4**(3), 393–404 (1981). DOI 10.1007/BF00304402

20. Maybury, M.: New Directions In Question Answering. In: T. Strzalkowski, S.M. Harabagiu (eds.) Advances in Open Domain Question Answering, *Text, Speech and Language Technology*, vol. 32, chap. New Direct. Springer Netherlands, Dordrecht (2008). DOI 10.1007/978-1-4020-4746-6. URL http://www.springerlink.com/index/10.1007/978-1-4020-4746-6
21. McGuinness, D.L.: Question answering on the semantic Web (2004). DOI 10.1109/MIS.2004.1265890. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1265890
22. Mendes, A.C., Coheur, L.: When the answer comes into question in question-answering: survey and open issues. Natural Language Engineering **19**(1), 1–32 (2013). DOI 10.1017/S1351324911000350. URL http://journals.cambridge.org/abstract{_}S1351324911000350
23. Miller, G.A.: WordNet: A Lexical Database for English. Communications of the ACM **38**(11), 39–41 (1995)
24. Moriceau, V.: Numerical data integration for cooperative question-answering. In: European Chapter of the Association for Computational Linguistics Workshop On KRAQ Knowledge And Reasoning For Language Processing, pp. 42–49. Association for Computational Linguistics (2006). URL http://dl.acm.org/citation.cfm?id=1641493.1641501
25. Nivre, J.: Dependency grammar and dependency parsing. MSI report **5133**(1959), 1–32 (2005)
26. Papineni, K., Roukos, S., Ward, T., Zhu, W.J.: BLEU: a method for automatic evaluation of machine translation. In: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics - ACL '02, pp. 311–318. Association for Computational Linguistics, Morristown, NJ, USA (2001). DOI 10.3115/1073083.1073135. URL http://dl.acm.org/citation.cfm?id=1073083.1073135
27. Perera, R., Nand, P.: RealText cs - Corpus Based Domain Independent Content Selection Model. In: 26th IEEE International Conference on Tools with Artificial Intelligence. IEEE Press (2014)
28. Perera, R., Nand, P.: The Role of Linked Data in Content Selection. Trends in Artificial Intelligence **8862**, 573–586 (2014). DOI 10.1007/978-3-319-13560-1_46. URL http://dx.doi.org/10.1007/978-3-319-13560-1{_}46
29. Perera, R., Nand, P.: A Multi-strategy Approach for Lexicalizing Linked Open Data. Computational Linguistics and Intelligent Text Processing pp. 348–363 (2015). DOI 10.1007/978-3-319-18117-2_26. URL http://link.springer.com/chapter/10.1007/978-3-319-18117-2{_}26
30. Perera, R., Nand, P., Klette, G.: RealText lex : A Lexicalization Framework for Linked Open Data. In: Internationa Semantic Web Conference (ISWC) - Demonstration, pp. 1–4 (2015)
31. Porter, M.F.: Snowball: A language for stemming algorithms (2001)
32. Radev, D.R., Jing, H., Budzikowska, M.: Centroid-based summarization of multiple documents: sentence extraction, utility-based evaluation, and user studies. Information Processing & Management 40.6 (2004): 919-938. p. 10 (2000). DOI 10.1016/j.ipm.2003.10.006. URL http://arxiv.org/abs/cs/0005020
33. Santorini, B., Kroch, A.: The syntax of natural language: An online introduction using the Trees program (2007). URL http://www.ling.upenn.edu/{~}beatrice/syntax-textbook.
34. Vargas-Vera, M., Motta, E.: AQUA-ontology-based question answering system. In: Mexican International Conference on Artificial Intelligence. Springer-Verlag, Mexico City, Mexico (2004). URL http://link.springer.com/chapter/10.1007/978-3-540-24694-7_48
35. Webber, B., Gardent, C., Bos, J.: Position statement: Inference in question answering. Proceedings of LREC (2002)
36. Yu, H., Lee, M., Kaufman, D., Ely, J., Osheroff, J.A., Hripcsak, G., Cimino, J.: Development, implementation, and a cognitive evaluation of a definitional question answering system for physicians. Journal of Biomedical Informatics **40**, 236–251 (2007). DOI 10.1016/j.jbi.2007.03.002